Remarks:

Specification:

It is requested that the above amendment to the specification be entered. In the amended paragraph, the specification referred to a reference numeral (121) that was inadvertently placed on the informal drawings initially filed with the application. On May 8, 2002, formal drawings were filed in the application. The formal drawings do not contain the reference numeral 121, but rather use the reference numeral 105 uniformly throughout. Accordingly, Applicants have amended the specification to correspond to the formal drawings by calling out reference numeral 105 instead of reference numeral 121 when referring to the element "Query Executes on Database Tables" of FIG. 3.

Claims:

Claims 1–33 remain for consideration in this application, with claims 1, 6, 9, 14, 19, 24 and 29 being independent. In the Office Action dated September 1, 2004 ("OA"), claims 1–3, 6, 9–11, 14–16, 19–21, 24–26, and 29–31 were rejected under 35 U.S.C. § 102(e) as being anticipated by Chinnici et al., U.S. Patent Application Publication No. 2002/0188616. Claims 4, 7, 12, 17, 22, 27, and 32 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Chinnici in view of Steel, JR. et al., U.S. Patent Application Publication No. 2001/0056420. Finally, claims 5, 8, 13, 18, 23, 28, and 33 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Chinnici in view of Agesen, U.S. Patent No. 6,711,672. Applicants respectfully traverse these rejections.

Regarding the rejections based on 35 U.S.C. § 102, it should be noted that a "claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Broos. V. Union Oil Co. Of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). Turning initially to the rejection of claim 1, Applicants respectfully assert that none of the applied references discloses, either expressly or inherently, each element of the claim. Chinnici, for example, does not disclose the steps of "receiving queries in a query language" and "representing the queries in accordance with a declarative language paradigm."

In rejecting claim 1, the Examiner cites paragraphs 38–39 of Chinnici. Particularly, the Examiner states that the language of "a rowset is a database structure containing information that represents a row or multiple rows of a table of the database" discloses "representing the queries in accordance with a declarative language paradigm." Applicants respectfully disagree. Representing queries in accordance with a declarative language paradigm involves, for example, translating or compiling a query "to yield an intermediate representation of the query . . . corresponding to the functional [i.e., declarative] language paradigm." (Application, page 13, lines 11–13.) In contrast, Chinnici explains that the rowset referred to by the Examiner is simply a subset of database rows (Chinnici, ¶ 38), which is utterly unrelated to a process of representing a query in accordance with a declarative language paradigm.

While Chinnici discloses converting the SQL command or rowset change "to a programming language call" (Chinnici, ¶ 38), this disclosure wholly omits the steps of "representing the queries in accordance with a declarative language paradigm, and converting the queries represented in a declarative language paradigm to an imperative language." Implementing these steps is neither an arbitrary design choice nor a trivial matter. (*See, e.g.*, application page 10, line 10 through page 11, line 4.) Declarative programming languages, it will be appreciated, are derived from Lambda calculus type representations wherein the model of computation is based on a system where relationships are specified directly in terms of the input data. Thus, declarative language programs are made up of sets of definitions or equations describing relations which specify *what* is to be computed rather than *how* it is to be computed. (Application, page 8, lines 20–30.) Imperative programming languages, in contrast, are derived from "the von Neumann model of store, an arithmetic logic unit, data, and instructions." (Application, page 9, lines 1–7) Imperative language programs, therefore, include step-by-step sequences of commands and explicitly state *how* the result is to be obtained. (Id.) The two types of programming languages are further distinguished in the application at page 8, line 20 through page 9, line 7 and page 11, lines 6–19.

As observed by the Examiner, Chinnici discloses converting "the request for the employee's information record to an EJB [Enterprise Java Beans] method for obtaining the record from the SQL server." (Chinnici, ¶ 39.) Java™ is an *imperative* language, not a *declarative* language. As explained above, the difference is significant. Thus, Chinnici wholly omits the steps of "receiving queries in a query language" and "representing the queries in accordance with a declarative language paradigm" and does not anticipate or render obvious claim 1.

Furthermore, Chinnici is completely devoid of any teachings that suggest a motivation to modify Chinnici to include the combination of elements of claim 1. Chinnici does not suggest a motivation, for example, to represent "the queries in accordance with a declarative language paradigm." In fact, Chinnici discourages use of a declarative language paradigm by stating that "[t]he Java™ programming language [an imperative lánguage] is suited for use in a three tier environment due to programming conventions such as Java Beans (JB) and Enterprise Java Beans (EJB)." (Chinnici, ¶ 10.) Thus, Chinnici teaches away from making modifications to include the elements of claim 1.

Claims that depend from claim 1 further include limitations that distinguish them from the prior art. Claim 2, for example, includes "converting the query language to an intermediate declarative representative, and thereafter converting the query to an imperative language representation of the queries and executing the imperative language queries." Chinnici wholly omits any reference to an intermediate declarative representation. Paragraph 54 of Chinnici, for example, discloses converting Visual Basic commands to SQL commands or their equivalents, and converting the SQL commands to EJB methods. Note that Visual Basic™ and Java™ are *imperative*, not *declarative*, programming languages.

Claims 6, 14, 19, 24, and 29 are rejected on the same grounds as claim 1. Therefore, the arguments set forth above in relation to claim 1 also apply to those claims. Furthermore, the argument set forth above in relation to claim 2 also applies to claims 10, 15, 20, 25 and 30. The remaining claims depend from those already discussed.

In view of the foregoing, a Notice of Allowance appears to be in order and such is courteously solicited.

The undersigned does not believe that any additional fees are due. However, should any additional fees be due, please charge them to Deposit Account No. 09-0460.

Respectfully Submitted,

By [illegible]

Jason E. Gorden, Reg. No. 46,734
HOVEY WILLIAMS LLP
2405 Grand Boulevard, Suite 400
Kansas City, Missouri 64108

ATTORNEYS FOR APPLICANTS